IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Application of:	Group Art Unit: 2167
Bobrovskiy, Stanislav M., et al.	Confirmation No. 3079
Application No.: 10/736,452	Examiner: Lu, Kuen S.
Filed: December 14, 2003	
For: METHOD AND APPARATUS FOR BUFFERING STREAMING MEDIA	

PRE-APPEAL BRIEF REQUEST FOR REVIEW

TO THE COMMISSIONER FOR PATENTS:

In response to the Final Office Action dated March 22, 2007 ("Office Action"), Applicant requests review of the final rejection in the above-identified application.

Listing of the Claims begin on page 2 of this paper.

Remarks begin on page 7 of this paper.

LISTING OF CLAIMS

1. (Previously Presented) A method of storing streamed presentation data within a container file, the method comprising:

receiving one or more data streams from each of one or more presentation sources within the presentation;

creating within the container file, a virtual file for each of the one or more presentation sources;

temporarily storing first data associated with a first data stream of a first presentation source in association with a first virtual file corresponding to the presentation source;

determining a container file size of the container file;

temporarily storing additional data from the first data stream in place of at least a portion of the first data if the container file size is within a predetermined range of an identified maximum buffer size; and

rendering at least one of said one or more data streams.

2. (Original) The method of claim 1, wherein the additional data from the first data stream is stored in place of at least a portion of the first data if the container file size is equal to or exceeds the identified maximum buffer size.

3. (Original) The method of claim 1, further comprising: temporarily storing second data associated with a second data stream of the first presentation source in association with the first virtual file; and temporarily storing additional data from the second data stream in place of at least a portion of the second data stored in association with the first virtual file if the container file size is within the predetermined range of the identified maximum buffer size.

4. (Original) The method of claim 3, further comprising: rendering one of the first and second data streams in real-time contemporaneous with the storing of at least one of the first and second data streams.

5. (Original) The method of claim 3, further comprising: temporarily storing data associated with a third data stream of a second presentation source in association with a second virtual

file; and temporarily storing additional data from the third data stream in place of at least a portion of the data stored in association with the second virtual file if the container file size is within the predetermined range of the identified maximum buffer size.

6. (Original) The method of claim 1, wherein the maximum buffer size is proportional to an amount of time indicated via a user interface.

7. (Original) The method of claim 1, wherein the maximum buffer size is dynamically increased during the storing of data from the first data stream.

8. (Original) The method of claim 1, wherein the first data and additional data are stored in a native packet format prior to a decoding process.

9. (Original) The method of claim 1, wherein each virtual file comprises: at least a first data block; and a file descriptor block containing at least a seek index and a seek index granularity, wherein the seek index indicates a plurality of equally distributed data blocks within the corresponding virtual file and the granularity indicates a size for each of the data blocks.

10. (Original) The method of claim 9, wherein the additional data is stored in place of the first data beginning with the first data block and continuing with successive data blocks of the first virtual file.

11. (Original) The method of claim 9, wherein if the container file size is within the predetermined range of the identified maximum buffer size, the seek index granularity is increased so as to increase data block size without changing the number of seek index entries.

12. (Original) The method of claim 9, further comprising: receiving a user indication identifying a location corresponding to a time (T) within the presentation; identifying a seek position for each virtual file, wherein each seek position is determined by dividing time (T) by the seek granularity for the corresponding virtual file; and contemporaneously rendering in real-time, data stored in each virtual file at the respective seek positions.

13. (Previously Presented) A machine readable medium having machine executable instructions, which when executed operate to implement a method comprising:

receiving one or more data streams from each of one or more presentation sources within a presentation;

creating within a container file, a virtual file for each of the one or more presentation sources;

temporarily storing first data associated with a first data stream of a first presentation source in association with a first virtual file corresponding to the presentation source;

determining a container file size of the container file;

temporarily storing additional data from the first data stream in place of at least a portion of the first data if the container file size is within a predetermined range of an identified maximum buffer size; and

rendering at least one of said one or more data streams.

14. (Original) The machine readable medium of claim 13, wherein the additional data from the first data stream is stored in place of at least a portion of the first data if the container file size is equal to or exceeds the identified maximum buffer size.

15. (Original) The machine readable medium of claim 13, further comprising instructions to temporarily store second data associated with a second data stream of the first presentation source in association with the first virtual file; and temporarily store additional data from the second data stream in place of at least a portion of the second data stored in association with the first virtual file if the container file size is within the predetermined range of the identified maximum buffer size.

16. (Original) The machine readable medium of claim 15, further comprising instructions to render one of the first and second data streams in real-time contemporaneous with the storing of at least one of the first and second data streams.

17. (Original) The machine readable medium of claim 15, further comprising instructions to: temporarily store data associated with a third data stream of a second presentation source in association with a second virtual file; and temporarily store additional data from the third

data stream in place of at least a portion of the data stored in association with the second virtual file if the container file size is within the predetermined range of the identified maximum buffer size.

18. (Original) The machine readable medium of claim 13, wherein the maximum buffer size is proportional to an amount of time indicated via a user interface.

19. (Original) The machine readable medium of claim 13, wherein the maximum buffer size is dynamically increased during the storing of data from the first data stream.

20. (Original) The machine readable medium of claim 13, wherein the first data and additional data are stored in a native packet format prior to a decoding process.

21. (Original) The machine readable medium of claim 13, wherein each virtual file comprises: at least a first data block; and a file descriptor block containing at least a seek index and a seek index granularity, wherein the seek index indicates a plurality of equally distributed data blocks within the corresponding virtual file and the granularity indicates a size for each of the data blocks.

22. (Original) The machine readable medium of claim 21, wherein the additional data is stored in place of the first data beginning with the first data block and continuing with successive data blocks of the first virtual file.

23. (Original) The machine readable medium of claim 21, wherein if the container file size is within the predetermined range of the identified maximum buffer size, the seek index granularity is increased so as to increase data block size without changing the number of seek index entries.

24. (Original) The machine readable medium of claim 21, further comprising instructions to receive a user indication identifying a location corresponding to a time (T) within the presentation; identify a seek position for each virtual file, wherein each seek position is determined by dividing time (T) by the seek granularity for the corresponding virtual file;

and contemporaneously render in real-time, data stored in each virtual file at the respective seek positions.

REMARKS/ARGUMENTS

35 U.S.C. § 103 Rejections

Applicants respectfully submit that the § 103 rejections in the Office Action are based on clearly erroneous errors, including the error that a "container file" as recited in Claims 1 and 13 is the equivalent of a hard drive with a File Allocation Table ("FAT"). In addition, the stated motivation to combine the cited references was clearly erroneous. These clear errors lead the Examiner to incorrectly conclude that Claims 1-11 and 13-23 are unpatentable over Published U.S. Patent application No. 2003/0110504 to Plourde et al. (hereinafter "*Plourde*") in view of Korst's U.S. Patent No. 6,205,525 (hereinafter "*Korst*").

A "container file" is not a hard drive with a FAT.

Claim 1 recites as follows:

> 1. A method of storing streamed presentation data within a container file, the method comprising:
> receiving one or more data streams from each of one or more presentation sources within the presentation;
> creating within the container file, a virtual file for each of the one or more presentation sources;
> temporarily storing first data associated with a first data stream of a first presentation source in association with a first virtual file corresponding to the presentation source;
> **determining a container file size of the container file;**
> temporarily storing additional data from the first data stream in place of at least a portion of the first data if the container file size is within a predetermined range of an identified maximum buffer size; and
> rendering at least one of said one or more data streams.

Several attributes of a "container file" are apparent from a close reading of Claim 1. First, a "container file" must vary in size, otherwise there would be no need to "determine a container file size of the container file." Second, the contents of a container file, the "virtual file for each of the one or more presentation sources," are dynamic and transient, storing **temporarily** data from a data stream until it can be rendered.

Plourde, by contrast, discloses a Digital Video Recorder ("DVR") system that allows the long term storage of subscriber television content on a storage device, preferably a hard drive (*Plourde* para 87). *Plourde* teaches that the storage device is of a fixed size and that the

operating system makes use of a File Allocation Table ("FAT") to store information about the hard disk clusters and the files associated with those clusters (*Plourde* para 88).

On page 3, the Office Action suggests that a "container file," as in Claim 1, is taught by *Plourde's* hard drive with a FAT file that contains information about media content instance files. However, a container file, as in Claim 1, offers a level of flexibility that is not needed, nor taught or suggested by *Plourde's* hard drive with a FAT file. As demonstrated by the attributes discussed above, the size of a container file may be adapted to suit various circumstances, while a hard drive is not flexible enough to do so. In addition, the creation of *Plourde's* hard drive with a FAT file typically requires destructively "formatting" the hard drive. By contrast, container files can be easily and nondestructively created and disposed of as needed. Furthermore, because a FAT is a type of file system, *Plourde's* hard drive would typically involve a file system's overhead and would be complex to design and administer. By contrast, "container files" offer facilities to perform few complex tasks and are therefore not only more flexible than *Plourde's* hard drive, but also much simpler. Accordingly, it is clear error to say that *Plourde* teaches this element of Claims 1 and 13.

The Office Action further suggests that a "virtual file for each of one or more presentation sources," as in Claim 1, is taught by *Plourde's* media content instance files. In support of this suggestion, the Office Action on page 11 states that "the FAT [has] entries describing attributes of content media instance files where directory structured virtual file contains one or more entries and the structure does teach one directory having one entry." Apparently, the argument is that a "virtual file for each of one or more presentation sources" is identical to a disk drive with a file allocation table that contains an entry describing a single file. Applicants respectfully submit that the "virtual file for each of one or more presentation sources" is much more flexible and useful than the scenario described in the Office Action. Applicants further respectfully submit that *Plourde* lacks any suggestion that media content instance files may be virtually aggregated and organized by presentation source. Such convenience is offered only by Claims 1 and 13; therefore, it is clear error to say that *Plourde* teaches this element of these claims.

The clear errors, however, are not limited to those aspects of the Claims said to be taught by *Plourde*. If anything, the errors regarding *Korst* are even clearer. For example, after correctly noting that *Plourde* does not teach "determining a container file size," the Office

Action suggests on pages 3-4 that "determining a container file size," as in Claim 1, is taught by *Korst's* calculation of the size of the data block to be read in the following sweep operation. However, while *Korst* may disclose a process that involves a "determination" of a "size," even a cursory reading of *Korst* makes clear that it does not teach "determining a *container file* size." A brief summary of *Korst* explains why.

Korst discloses a video on demand *server* that efficiently buffers content data for streaming to a user device. The content data addressed in *Korst* is stored on a storage medium, such as a hard drive, and must be retrieved from the storage medium in blocks (*Korst* col. 7, lines 13-42). The calculation that *Korst* teaches involves figuring out how many blocks of data to read from the hard drive and insert into a buffer for ultimate delivery to a client device, a calculation that depends not on the contents of any container file, but on the number of streams that the server is sending out to client devices (*Korst* col. 7, lines 13-42). Thus, it is clear error to say that *Korst* teaches "determining a container file size" as in Claims 1 and 13 because unlike Claim 1, *Korst* **does not determine the size of any sort of container**.

Finally, it is clear error to say that "rendering at least one of said one or more data streams" is taught by *Korst's* scheduler's determining the number of active streams and calculating the size of the data block to be read, as suggested by the Office Action on page 4. The cited reference to *Korst* does not teach or even suggest the "rendering" of a stream; rather, *Korst* teaches at most only that a data stream may be transmitted to a client, a process that is very different from "rendering" a data stream into a form that may be perceived by a user.

It was clear error to combine Plourde and *Korst.*

On page 10 of Applicants' response of October 17, 2006, Applicants noted that "the Examiner has attempted to use the pending application to define the problem to be solved by reference to different elements from the prior art." On page 12 of the Office Action, the Examiner replied that "the motivation or suggestion of combination of the references does come from the BACKGROUNDS OF INVENTION of the references...." Considering the context of and problems solved by *Korst* and *Plourde* it would clearly not have been obvious to combine the two references at least because such combination would not yield the results

obtained in the Claims. Hence, the combination of *Korst* and *Plourde* would not yield predictable results. In addition, the problems to be solved that are laid out in the background sections of *Korst* and *Plourde* relate at most only peripherally to the problems that the Applicants address in the recited claims. For these reasons, it is clear error to assert that one of ordinary skill in the art would have had any motivation to combine *Korst* and *Plourde*.

Specifically, *Plourde's* background section explains that the invention addresses the problems that arise when a television viewer wishes to watch two or more programs at the same time or wishes to watch a program that is on when the viewer is away from the television (*Plourde* para 5). *Plourde* solves these problems by providing a better "buffering" mechanism that would be employed by a digital video recorder, such as a Tivo®. In other words, *Plourde* works with media data that arrives from elsewhere at a constant rate, at a known time, for a known duration.

By contrast, *Korst* explains that it addresses problems faced by streaming media **servers** that may have to stream many different streams of different pieces of media out to many different clients. To that end, *Korst* provides, "[t]o supply data to a user as a continuous data stream, special scheduling schemes for reading data from the disks are required with an appropriate scheme for temporarily buffering the read data before the data is supplied to the user" (*Korst* col. 1, lines 55-58). In other words, *Korst* works with a large quantity of media data that is stored locally, but that distant, disparate clients may want to access at unknown rates, at unknown times, for unknown durations.

Thus, as their background sections make clear, the **only** commonalities between *Korst* and *Plourde* are that they both relate to media data sent from one place to another. On the other hand, there are many differences that would have deterred one of ordinary skill in the art from combining the two. Particularly, *Korst* addresses problems that are not faced by media-receiving client devices, such as are the subject of *Plourde*. Unlike streaming media servers, which are the subject of the problems solved by *Korst,* a client device has no need to read data in sweeps from a hard drive; a client device does not need to service multiple requests for data streams; nor does a client device need to juggle the delivery of different pieces of media that are stored in disjoint sectors on a local hard drive. Clearly, the problems faced by streaming servers differ greatly from those faced by client devices. Therefore, there is no reason why a person of ordinary skill in the art would have been motivated to combine

Plourde, which deals with client devices, and *Korst*, which deals with server devices. Given these differences, it is clear error to say that one would have had any motivation to combine *Korst* and *Plourde*.

When the clear errors discussed above are corrected, the recited claims cannot be said to be obvious in light of *Korst* and *Plourde*. Applicants therefore submit that all pending claims are in condition for allowance. Applicants respectfully request reconsideration and withdrawal of the rejection of amended Claims 1 and 13. In addition, Applicants suggest that Claims 2-11 and 14-23, which depend directly or indirectly on Claims 1, and 13, are patentably distinct over the combination of *Plourde* in view of *Korst* and are also in condition for allowance.

CONCLUSION

Applicant submits that all pending claims are in condition for allowance. Accordingly, early and favorable action allowing all of the pending claims and passing this application to issue is respectfully requested. The Examiner is respectfully requested to contact the undersigned at the telephone number below if there are any remaining questions regarding this application.

We believe the appropriate fees accompany this transmission. If, however, insufficient fee payment or fee overpayment occurs, the amount may be withdrawn or deposited from/to Axios Law Group's deposit account. The deposit account number is 50-4051.

Respectfully submitted,
AXIOS LAW GROUP

Date: June 20, 2007

by: /Adam L.K. Philipp/
Adam L.K. Philipp
Reg. No.: 42,071

AXIOS Law Group
1725 Westlake Avenue N, Suite 150
Seattle, WA 98109
Telephone: 206-217-2200